An AC-3/MPEG multi-standard audio decoder IC

Li, S. Rowlands, J. Ng, P. Gill, M. Youm, D.S. Kam, D. Song, S.W. Look, P.
Dept. of Digital Compression Products, Texas Instrum. Inc., Dallas, TX, USA;

This paper appears in: **Custom Integrated Circuits Conference, 1997., Pı of the IEEE 1997**

Meeting Date: 05/05/1997 - 05/08/1997
Publication Date: 5-8 May 1997
Location: Santa Clara, CA USA
On page(s): 245 - 248
Reference Cited: 2
Number of Pages: 606
Inspec Accession Number: 5730286

Abstract:
The emerging digital audio compression technology brings both an opportunit challenge to IC design. High quality multichannel audio is quickly becoming aı indispensable part of an entertainment system. The algorithms used in the co technology result in complex VLSI ICs. The work presented in this paper is ab design of a dedicated, high precision, and low cost **AC3/MPEG** multi-standarc **decoder**. The audio IC's hardware and software architecture, as well as desiç simulation/verification methodology are discussed in detail

Index Terms:
VLSI audio coding code standards data compression decoding digital signal proce AC-3/**MPEG** multi-standard audio **decoder** IC VLSI algorithm design digital audio c entertainment system multichannel audio simulation verification

Documents that cite this document
There are no citing documents available in IEEE Xplore at this time.

An AC-3/MPEG Multi-standard Audio Decoder IC

Stephen Li, Jon Rowlands, Pius Ng, Maria Gill, D.S. Youm
David Kam, S.W. Song, Paul Look

Digital Compression Products
Texas Instruments Incorporated
Dallas Texas 75265

Abstract

The emerging digital audio compression technology brings both an opportunity and a new challenge to IC design. High quality multichannel audio is quickly becoming an indispensable part of an entertainment system. The algorithms used in the compression technology result in complex VLSI IC's. The work presented in this paper is about the design of a dedicated, high precision, and low cost AC3/MPEG multi-standard audio decoder. The audio IC's hardware and software architecture, as well as design and simulation/verification methodology are discussed in detail.

Introduction

Two of the audio compression standards that are being widely adopted are the Dolby Laboratories' AC-3 and ISO's MPEG. The AC-3 standard has been adopted for use on laser disc, DVD, the US ATV system, and some emerging digital cable systems. The MPEG standard has gained wide acceptance in satellite broadcasting, CD-ROM publishing, and DAB. The two standards potentially have a large overlap of application areas.

Both of the compression standards are based on psycho-acoustics of the human perception system [1][2]. The input digital audio signals, PCM, are split into frequency subbands using an analysis filter bank. The subband filter outputs are then downsampled and quantized using dynamic bit allocation in such a way that the quantization noise is masked by the sound and remains imperceptible. These quantized and coded samples are then packed into audio frames that conform to the respective standard's formatting requirements. For a 5.1 channel system, high quality audio can be obtained for compression ratio in the range of 10:1.

Both of the standards are capable of carrying up to 5.1 channels of audio data and incorporate a number of variants including sampling frequencies, bit rates, speaker configurations, and a variety of control features. However, the standards differ in their bit allocation algorithms, transform length, control feature sets, and syntax formats.

The decoder IC fully complies with the ATSC AC-3 and ISO MPEG-1 standard. It accepts all AC-3 or MPEG compliant audio data streams and produces two-channel PCM output. AC-3 input bit streams with more than two channels are downmixed to produce two output channels. It fully supports dynamic range compression, dialog normalization, and all operational modes including those for karaoke. In the case of MPEG-2 audio, the stereo MPEG-1 compatible signal is decoded and presented over the two-channel PCM output. In addition, the IC accepts up to 8 channels of PCM data and produces two channels output using user-supplied downmixing coefficients. In all cases, the decoded PCM can be output in 16, 20, or 24-bit format.

Device Architecture

The architectural hardware and software implementation reflect the two very different kinds of tasks to be performed by the decoder IC. First is the front-end decoding part. Here it must unpack the variable length encoded pieces of information from the bitstream. Additional decoding results in a set of frequency coefficients. The second part is the synthesis filter bank that converts the frequency domain coefficients to PCM data. In addition, the IC also needs to support dynamic range compression, downmixing, error detection and concealment, time synchronization, and other system resources allocation and management functions.

The architectural decision took into consideration factors such as flexibility, chip area, cost and performance, which led to the design of a dual-processor architecture.

Figure 1 is a functional block diagram of the audio decoder IC. The design is composed of two autonomous processing units working together through shared memory supported by multiple I/O modules. The operation of each unit is data-driven. The synchronization is carried out by the Central Processing Unit (CPU) which acts as the master processor.

0-7803-3669-0 $5.00 © 1997 IEEE

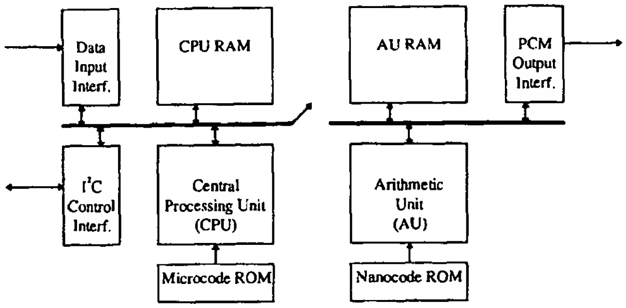

Figure 1 Functional Block Diagram

A typical operation cycle is as follows: Coded data arrives at the Data Input Interface asynchronous to the decoder IC system clock. The Data Input Interface synchronizes the incoming data to the 27 MHz decoder processing clock and transfers the data to CPU memory through DMA. The CPU reads the compressed data from the buffer, performs various decoding operations, and writes the unpacked frequency domain coefficients to the AU RAM, a shared memory between CPU and AU. The Arithmetic Unit is then activated and performs subband synthesis filtering, which produces the reconstructed PCM samples. The PCM Output Interface takes PCM samples from AU RAM through DMA and then formats and outputs them to an external D/A converter. Additional functions performed by the CPU includes control and status I/O, as well as overall system resource management.

The CPU is a programmable processor with hardware acceleration and instructions customized for audio decoding. It is a 16-bit RISC processor with register-to-register operations and an address generation unit operating in parallel. This unit is capable of performing an ALU operation, a memory I/O, and a memory address update operation in one system clock cycle. Three addressing modes: direct, indirect, and registered are supported. Selective acceleration is provided for field extraction and buffer management to reduce control software overhead. Figure 2 is a list of the instruction set and Figure 3 shows a block diagram of its architecture.

Instruction Mnemonics	Functional Description
Move	Register move
And	Logical and
Or	Logical or
cSat	Conditional saturation
Ash	Arithmetic shift
LSh	Logical shift
RoRC	Rotate right with carry
GBF	Get bit-field
PBF	Pack bit-field
Add	Add
AddC	Add with carry
cAdd	Conditional add
Xor	Logical exclusive or
Sub	Subtract
SubB	Subtract with borrow
SubR	Subtract reversed
Neg	2's complement
cNeg	Conditional 2's complement
Bcc	Conditional branch
DBcc	Decrement & conditional branch
IOrd	IO reg to memory move
IOwr	Memory to IO reg move
auOp	AU operation
Sleep	Power down unit

Figure 2: CPU instruction set

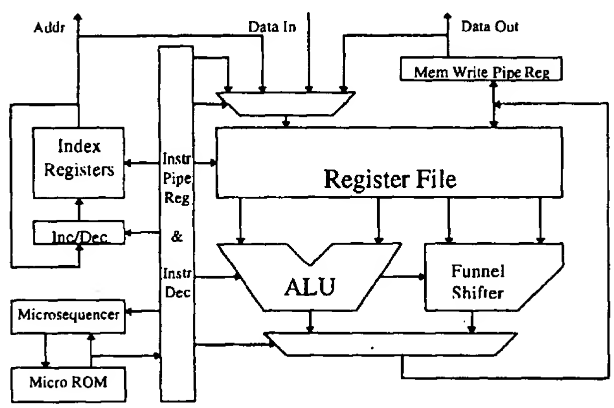

Figure 3: CPU architecture

The unit has two pipeline stages: Instruction Fetch/Predecode, and Decode/Execution. The decoding is split and merged with the Instruction Fetch and Execution respectively. This arrangement reduces one pipeline stage and thus branching overhead. Also, the shallow pipe operation enables the processor to have a very small register file (three general purpose registers, a dedicated bitstream address pointer, and a control/status register) since memory can be accessed with only a single cycle delay.

The Arithmetic unit is a programmable fixed point math processor that performs the subband synthesis filtering. The module receives frequency domain coefficients from the CPU by means of the shared AU memory. After the CPU has written a block of coefficients into the AU memory, it activates the AU through a coprocessor instruction. The CPU is then free to continue decoding the audio input data.

Synchronization of the two processors is achieved through interrupts.

The width of the datapath in the arithmetic unit was chosen so that the resulting PCM audio will be of superior quality after processing. The width was determined by comparing the results of fixed point simulations to the results of a similar simulation using double-precision floating point arithmetic. In addition, double-precision multiplies are performed selectively in critical areas within the subband synthesis filtering process.

Since the product is targeted toward a consumer market, careful consideration has been given to power management. The AU powers up when a coprocessor instruction is issued by the CPU and powers down after execution. The CPU is designed to be capable of decoding the worst case frame, so it has a lot of spare cycles for an average one. When there are no more active tasks, the kernel issues the Sleep instruction to power down the CPU. This power-on-demand mechanism successfully meets the design goal of worst case processing and average case power saving.

Software Architecture

Each hardware component in the audio decoder IC has an associated software component, including the compressed bitstream input, audio sample output, host command interface, and the audio algorithms themselves. These components are overseen by a kernel that provides real-time operation using interrupts and software multi-tasking. The software was developed in microcode using proprietary tools.

The software architecture block diagram is shown in Figure 4. Each of the blocks corresponds to one system software task. These tasks run concurrently and communicate via global memory. They are scheduled according to priority, data availability, and synchronized to hardware using interrupts. The concurrent data-driven model reduces RAM storage by allowing the size of a unit of data processed to be chosen independently for each task.

The software operates as follows. The Data Input Interface buffers input data and regulates flow between the external source and the internal decoding tasks. The Transport Decoder strips out packet information from the input data and emits a raw AC-3 or MPEG audio bitstream, which is processed by the Audio Decoder. The PCM Output Interface synchronizes the audio data output to a system-wide absolute time reference and, when necessary, attempts to conceal bitstream errors. The I^{2}C Control Interface accepts configuration commands from an external host and reports device status. Finally, the Kernel responds to hardware interrupts and schedules task execution.

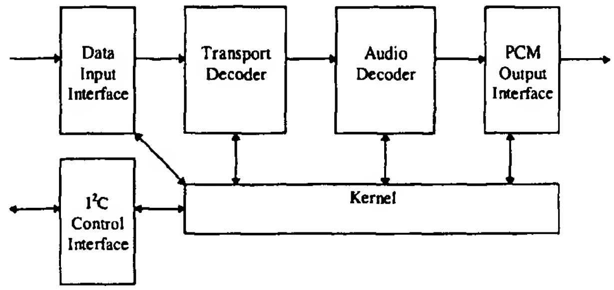

Figure 4: Software architecture

The audio decoder IC is a real-time device, so the software has both logical correctness and strict timing deadline requirements. These requirements were verified using a combination of analysis and emulation.

The Transport Decoder and Audio Decoder tasks were tested off-line for logical correctness against C language algorithm models. A large suite of test data was available for comparison at multiple internal points of the decoding algorithms, and was compared to the microcode results in emulation. The emulation system collected the final PCM output in bulk, allowing automated regression testing.

Real-time correctness must be verified for the individual tasks and, ideally, for all possible interactions of the tasks. This requirement is complicated by real-time interrupts. The individual tasks were verified by emulation and worst case analysis where appropriate. The formalism provided by the Kernel made worst case analysis of the interactions tractable. The analysis itself was checked using emulation.

Simulation Environment

Decoding an AC-3 compressed frame requires around 133,000 system clock cycles. Meanwhile, the performance of the Synopsys VSS event driven VHDL simulator for this design is in the order of less than 10 cycles per second (cps) on a Sun Sparc 20 workstation. Therefore, it takes 3.7 hours to decode a compressed frame. Even with the latest IKOS NSIM hardware accelerator, the performance only improved to 100+ cps. Dolby's AC-3 test suite contains more than 2000 compressed frames, which would take 11 months of continuous execution to decode with a single workstation. In addition, to verify features such as error concealment, complex streams of numerous frames are required. Although this level of verification may be done by using the actual silicon, debugging in silicon with the complexity of this design is difficult. The time-to-market as well as the number of silicon revisions required are significant concerns in the consumer market. Thus, an emulation environment was adopted as the primary development and debugging tool for this design. In order to achieve this objective using Quickturn Design System's Enterprise emulation system, the team developed an environment similar to a conventional software simulator

with features such as single stepping, breakpoints, monitors, and testing environment controls.

With this environment, the hardware design team used the software simulator to verify the initial reset sequence and module level testing. In the meantime, the design was synthesized without timing constraints so that it could be compiled into the emulator as soon as possible. Then, the software team used the emulator to develop software and debug hardware. Primitive disassembler and data capture capability were developed for this emulation environment to assist the software debugging process. Since the emulator was running at 500kHz, regression of the entire Dolby test suite took about six hours. The software team developed and debugged the design during the day and regression was carried out at night to ensure the integrity of the design changes.

The design was taped out after a sign-off sequence consisting of functional verification using emulation, timing verification using IKOS NSIM, and layout to schematic verification. With this preparation, the design achieved all functional requirements in first pass silicon. Although the emulation effort required a full-time designer for improvement and maintenance, the first pass success proved its value and capability. The design flow is shown in Figure 5.

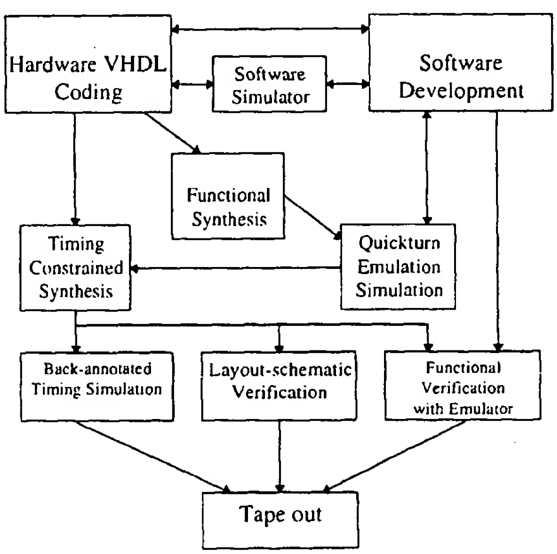

Figure 5: Design flow

Conclusion

Figure 6 is a photo of the finished chip. The design is implemented using Texas Instruments' TEC3000T CMOS Gate Array with embedded memory modules. The total gate count of the entire IC is about 30,000. The CPU and AU have approximately 8,000 gates each. The embedded software consists of about 6,000 lines of microcode. The IC fully met all its functional requirements in its first pass silicon. The key to the success of this project is the top-down design methodology, well-coordinated software/hardware co-development, and the implementation of a successful simulation/emulation strategy.

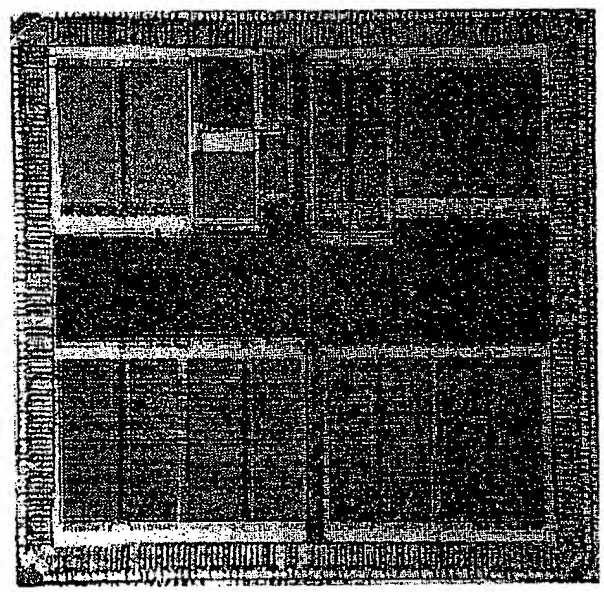

Figure 6: Audio decoder IC layout

References

[1] United States Advanced Television Systems Committee "Digital Audio Compression (AC-3) ATSC Standard," Doc. A/52, Nov., 1994

[2] ISO Standard "Coding of Moving Pictures and Associated Audio for Digital Storage Media at up to about 1.5M Bit/s," CD 11172-3, July, 1992.